VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Absolvování individuální odborné praxe
Individual Professional Practise in the Company

2010 **Lukáš Révay**

Prehlásenie

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne. Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

Dňa 6.5.2010 v Ostrave …....

podpis

Poďakovanie

Chcel by som poďakovať firme Tieto za umožnenie vykonania odbornej bakalárskej praxe a môjmu konzultantovi Ing. Zdeňkovi Šmídovi za jeho trpezlivosť a ústretovosť pri vedení odbornej praxe.

Ďalej by som rád poďakoval zamestnancom, s ktorými som spolupracoval za ich odborné rady, pomoc a kolegialitu pri riešených úlohách.

Abstrakt

Práca pojednáva o absolvovaní odbornej praxe vo firme Tieto. Cieľom je popísať zadané úlohy a ich riešenie. Bakalárska práca sa taktiež zaoberá popisom získaných praktických skúseností a znalostí, ktoré som nadobudol, alebo tých, ktoré mi chýbali.

Kľúčové slová

Odborná prax, získané skúsenosti a znalostí, C/C++, Qt framework, sférická trigonometria, GPS, NMEA, Nokia, Maemo, Windows Mobile, Tieto

Abstract

The work deals with professional experience in the company Tieto. The aim is to describe the assigned tasks and their solutions. Bachelor thesis also describes gained practical experience and knowledge I have acquired, or those I missed.

Keywords

Individual practise, gained experiences and knowledges, C/C++, Qt framework, spherical trigonometry, GPS, NMEA, Nokia, Maemo, Windows Mobile, Tieto

Zoznam použitých symbolov a skratiek

R&D	research and development
GPS	Global Positioning System
NMEA	National Marine Electric Association
ASCII	American Standard Code for Information Interchange
HTTP	Hypertext Transfer Protocol
URL	Uniform Resource Locator
UI	User Interface
TCP	Transmision Control Protocol
API	Application Programming Interface

Obsah

1. Úvod

Celá bakalárska práca pojednáva o absolvovaní odbornej individuálnej praxe vo firme Tieto. Touto prácou sa snažím popísať všetky úlohy, ktoré mi boli zadané a mal som ich za úlohu vyriešiť. Taktiež popisujem aj postup riešení daných úloh.

Po absolvovaní výberového konania formou pohovoru ma čakalo zoznámenie sa s chodom spoločností, kde počas dvoch dní som bol oboznámený s bezpečnosťou, duchovným vlastníctvom, ako aj s korporatívnymi obmedzeniami firmy.

Nasledovalo zaradenie do oddelenia R&D, kde som bol umiestnený do teamu, ktorý navrhoval GPS hru pre mobilné zariadenia s operačnými systémami Windows Mobile a Maemo. V tomto tíme som pôsobil ako junior developer.

Mojou úlohou bol návrh UI na strane klienta. Konkrétne išlo o implementáciu kompasu na určovanie smeru pohybu užívateľa aj so zobrazovaním potrebných veličín pre zemepisnú orientáciu. Nešlo však len o užívateľské rozhranie, ale aj o logiku, ktorá by bola schopná reprezentovať a prepočítavať parametre vhodné na zobrazovanie v UI.

Keďže k určovaniu presnej polohy nebol kompas vhodným kandidátom dostal som za úlohu zobrazovať polohu užívateľa na mape aj s príslušnými cieľmi. Pri riešení tohto problému som si musel naštudovať aj niečo s kartografie.Pretože bolo potrebné tieto komponenty aj otestovať v teréne tak som počas obednej prestávky testoval túto aplikáciu na mobilnom zariadení s operačným systémom Maemo.

Tento projekt však bol internou záležitosťou firmy a tá väčšinou pracuje na externých projektoch, preto som nezostal v tomto týme dlho, po troch mesiacoch som na tejto aplikácií začal pracovať sám až do konca svojej praxe. To mi však nebránilo v zdokonalení sa v jazyku C/C++ a Qt frameworku [5] vhodného pre návrh užívateľských rozhraní.

2. Popis odborného zamerania firmy Tieto

Tieto je IT spoločnosť poskytujúca IT, R&D a konzultačné služby. S približne 17 000 expertmi, je jednou z vedúcich IT spoločností v severnej Európe a celosvetový líder vo vybraných odvetviach.

Zaujíma sa o oblasti, kde má hlboké porozumenie s potrebami svojich zákazníkov. Zameriava sa hlavne na podporu trhov s veľkými a stredne-veľkými firmami v Severnej Európe, Nemecku a Rusku. A to v oblastiach telekomunikácií, lesov, ropného a plynárenského priemyslu ako aj digitálnych služieb. Spolupracuje s mnohými vedúcimi firmami a organizáciami.

Pretože má mnoho oddelení s rôznym zameraním, tak bude vhodné zamerať sa len na to ,v ktorom som pracoval počas svojej praxe. Oddelenie R&D je hlavným poskytovateľom výskumu a vývoja v niekoľkých odvetviach priemyslu vo viac ako tucte krajín. Zákazníkmi sú svetový lídri v oblastiach mobilných zariadení, telekomunikačnej infraštruktúry a automobilového priemyslu.

R&D sa delí na oddelenia :

- Automotive - profesionálny vývoj pre systémy a kompetencie v automobilovom priemysle a dopravnej telematike.
- Mobile Devices - vývoj softvéru pre mobilne zariadenia, produkty a nástroje potrebne v R&D
- R&D Networks - profesionálne služby pre celkový životný cyklus sieťovej infraštruktúry R&D, systémovú integráciu a nasadenie.

Odbornosť oddelenia R&D je veľmi univerzálna, pretože spolupracuje s vyspelými a rýchlo sa vyvíjajúcimi odvetviami patriacimi vláde, podnikom, či jednotlivým spotrebiteľom. Pre to aby sa dostavila táto úspešnosť vyvinula firma Tieto, metódy a procesy tak aby uspokojovali zákazníkov. Kompetencie tohto oddelenia sa tykajú hlavne toho s akými technológiami sa stretlo respektíve aké znalosti môže poskytnúť. Za spomenutie iste stoji multiplatformový vývoj s Qt frameworkom od Nokie, ako aj vývoj Embeeded Systémov, práca s operačnými systémami pre mobilné zariadenia ako Maemo, Symbian, Android a mnoho ďalších. Tým, že sa jedna o oddelenie, ktoré sa zaoberá inováciami a vývojom svedči aj fakt, že počas môjho pôsobenia som sa stretol s radou technológii a aplikácií, ktoré boli novinkami.

3. Popis aplikácie

Táto aplikácia je určená pre mobilné zariadenia s operačným systémom Windows Mobile(Microsoft), alebo Maemo(Debian) s použitím programovacieho jazyka C/C++ a Qt frameworku od Nokie. Použitím jazyka C/C++ sa síce nevyhneme spravovaniu pamäte a iným druhom spravovania, ale za túto cenu získame pomerne vyšší výkon aplikácie oproti konkurenčným jazykom C#, alebo Java. S použitím daného frameworku sa nám vývoj aplikácie rovnako uľahčí, pretože môžeme použiť mnoho, už pripravených balíkov.

Aplikácia je založená na hierarchií klient server. Server zabezpečuje komunikáciu medzi klientmi. Počúva na dvoch portoch, z ktorých je jeden na asynchrónne požiadavky od klientov a druhý na synchrónne dotazy posielané serverom. Server pracuje na vláknovom princípe. Klient je navrhnutý na komunikáciu s GPS modulom a získavaním informácií o polohe a ich následnou reprezentáciou, ako aj posielaním na server. Taktiež používa mapy, konkrétne Google maps na reprezentáciu polohy prijímača a cieľových bodov.

Obr. 1: Logo Qt frameworku

Obr. 2: Logo Maemo

Obr. 3: Logo Windows Mobile

4. Serializácia a deserializácia dát

Serializácia a deserializácia je implementovaná pomocou tried **obinstream** a **ibinstream**, ktoré sú obe potomkami std::ostream. Pretože pracujeme s dátovým tokom môžeme dáta fragmentovať a defragmentovať tým spôsobom, že kontrolujeme kde sa nachádzame v dátovom toku a koľko nám ostáva kým ho celý spracujeme. Tým dochádza k tomu, že môžeme posielať akékoľvek dlhé dáta.

Pri deserializácií dochádza k spetnému skladaniu dát a to spôsobom vytvárania objektov z dátového toku. To znamená, že na koniec každej časti sa nám pripojí nasledujúci fragment a my, len kontrolujeme, či súčet jednotlivých dĺžok odoslaných dát nepresahuje celkovú dĺžku. Vo väčšine prípadov serializujeme udalosti a odpovede na ne, či už na strane servera alebo klienta. Z tohto dôvodu sme implementovali triedu CEvent, v ktorej máme definovaný enum udalostí, ktoré môžu nastať medzi klientom a serverom. Táto trieda je schopná serializovať svoje inštancie na obinstream a spätne ich deserializovať na ibinstream.

5. Serverová časť

Serverová časť tejto aplikácie je napísaná v C/C++. Server pracuje ako vláknový server, kde pre každého prihláseného klienta sa vytvorí jedno vlákno. Počúva na dvoch portoch, aby bola dosiahnuté, čo najlepšie riešenie pri komunikácií medzi serverom a klientom.

5.1. *Sieťová vrstva*

Pretože chceme reagovať nezávisle na udalosti klientov a ich riadenie, otvorili sme si dva TCP porty na počúvanie 2358 a 2359.

Pri autentifikácií a ostatných úkonoch týkajúcich sa jedného klienta o tom neinformujeme ďalších klientov, iba posielame správu späť klientovi, ktorý poslal požiadavku, rozhodli sme sa použiť jeden port. Druhým portom posielame dáta vtedy, keď sa jedná o udalosti, ktoré sú posielané aj ostatným klientom, ako napríklad zmena polohy užívateľa, úspešné prihlásenie.

5.2. Transportná vrstva

Pri posielaní dát je zaručené ich kompletné odoslanie. Formát správy je nasledujúci. Číslo na začiatku správy udáva jej dĺžku a následne je predaný ukazateľ na dáta v pamäti. Aby nedochádzalo k posielaniu, už neplatných dát iným klientom, urobíme si preto kópiu odosielaných dát v pamäti a tie odošleme. Kopírovanú časť pamäte vymažeme.

Aby dochádzalo k správnej interpretácií dát a k správnej reprezentácií musia byť tieto u odosielateľa serializované a u príjemcu poskladané späť na pôvodné objekty, to znamená , že správa sa deserializuje. Serializácia a deserializácia boli popísané v kapitole 4 .

5.3. Spracovanie správ a udalostí

Spracovanie správ je založené na hierarchií udalostí a poslucháčov. Podľa toho, akého typu sú posielané dáta, je vytvorená inštancia objektov a ich následné spracovanie v jednotlivých triedach reprezentujúcich príslušné rozhranie poslucháča. V tomto prípade máme, len jedno rozhranie predstavujúce vrchol v hierarchií tried a to konkrétne IDataListener. Je to abstraktná trieda, ktorú je treba implementovať ak chceme prijímať dáta, ktoré nám budú posielané po sietí. V triede, ktorá nám reprezentuje pripojenie jednotlivých klientov udržiavame zoznam poslucháčov.

6. Klientská časť

Pretože na komunikáciu medzi serverom a klientom sa používajú rovnaké metódy a triedy na pársovanie správ, posielanie dát po sieti a nadväzovanie komunikácie nebudem toto opisovať pretože som základné princípy popísal už na strane servera.

Mojou úlohou bolo vytvoriť užívateľské rozhranie pre klienta. Pretože ide o hru s GPS bolo potrebne navrhnúť komponentu kompasu a mapy, tak aby bolo pre užívateľa jasné, kde sa nachádza a kam sa pohybuje respektíve aj to kde sa nachádzajú nejaké ciele.

Za návrhom užívateľského rozhrania sa však skrýva logika, ktorá pracuje so súradnicami ich reprezentáciou, počítaním vzdialeností, projekciou bodov a rátaním azimutov. Pri vykresľovaní mapy tiež dochádza k asynchrónnemu volaniu funkcií na serveri Google maps, ktorého odpoveďou je obrázok vo zvolenom formáte.

Následné vykresľovanie a práca s ním už prebieha v prostredí Qt, kde pre správne zobrazovanie je potrebné prepočítavať zemepisne súradnice na pixely. Je zrejme, že bez súradníc by to nešlo. Ich získavanie prebieha na klientovy cez GPS prijímač, kde je dôležité parsovať vety prijímača a z nich získavať súradnice prípadne iné informácie.

6.1. NMEA

NMEA[4] používa jednoduchý ASCII, sériový komunikačný protokol, ktorý definuje ako sú dáta prenášané z jedného vysielača na viacero prijímačov. Vysielač môže vytvárať jednosmernú komunikáciu s neobmedzeným počtom prijímačov a pritom môže využívať multiplexovanie. Pretože tento protokol používa sériový prenos je možné na počítači využiť port RS232, alebo USB pripojenie, ktoré je taktiež sériové. V našom prípade je použité bluetooth, ktoré je schopné emulovať sériový port. V mobilných telefónoch ktoré majú vstavané GPS prijímače, však toto nie je problém. Z hľadiska aplikácie je dôležité pársovanie jednotlivých viet posielaných pomocou tohto protokolu.

Formát viet:

- Každá veta začína znakom dolár ($).
- Nasledujúcich päť znakov identifikuje vysielač (dva znaky) a typ správy (tri znaky).

- Všetky dátové časti sú oddelené čiarkou.
- Kde nie sú žiadne dáta, tak prisluchajúca časť je prázdna.
- Prvý znak, ktorý nasleduje ihneď za poslednou časťou je hviezdička.
- Hviezdička je ihneď nasledovaná dvoma hexadecimálnymi číslicami reprezentujúcimi kontrolný súčet.
- Veta končí <CR><LF>.

Príklad NMEA vety: $GPAAM,A,A,0.10,N,WPTNME*32

Dôležité sú, len vety, z ktorých je možné zistiť geografickú polohu. Tieto vety začínajú $GPGGA alebo $GPVTG. Prvý typ veta slúži na určenie presnej 3D geografickej polohy, ako aj nadmorskej výšky, počtu satelitov a mnoho ďalších. Druhý typom určujeme azimut akým sa pohybujeme vzhľadom na severný geografický, alebo magnetický pól, akú rýchlosť pohybu dosahujeme atď. . V triede CNMEAParser sú zadefinované všetky metódy, ktoré sú potrebné na pársovanie viet prichádzajúcich z GPS prijímača. Po úspešnom parsovaní vety sa zavolá signál, ktorý je pripojený na sloty zodpovedné za ďalšiu prácu so súradnicami.

6.2. GPS formáty súradníc

Reprezentácia súradníc v rámci jednotlivých častí klienta sa takisto odlišuje. Iná je v prípadoch pri internom ukladaní v triede CGPSCoordinate a iná v prípadoch, keď sa posiela dotaz napríklad na Google maps api. Je teda žiadúce vykonávať rôzne konverzie pre reprezentácie vhodné v daných častiach programu. Interné uloženie súradníc je typu long, to znamená, že táto reprezentácia je v desaťtisícinach sekúnd.

Podľa toho, či sa daný bod nachádza vpravo alebo vľavo od poludníka, alebo pod a nad rovníkom sa mení znamienko súradníc. Pri parsovaní do tohto formátu sa ešte prepočítavajú jednotlivé časti súradníc, kde celé stupne sú vynásobené 60 000, k nim je prirátaná minútová časť vynásobená 60 a zostávajúce sekundy. Ak však chceme rátať vzdialenosti medzi bodmi a azimuty je treba prepočítavať tento formát do radiánov. Je to vhodnejšie hlavne pri dosadzovaní do goniometrických funkcií. Prepočet do radiánov sa urobí následovne. Reprezentáciu v celo-číselnom formáte prevedieme na stupne a tie následne na radiány, ktoré sú typu double.

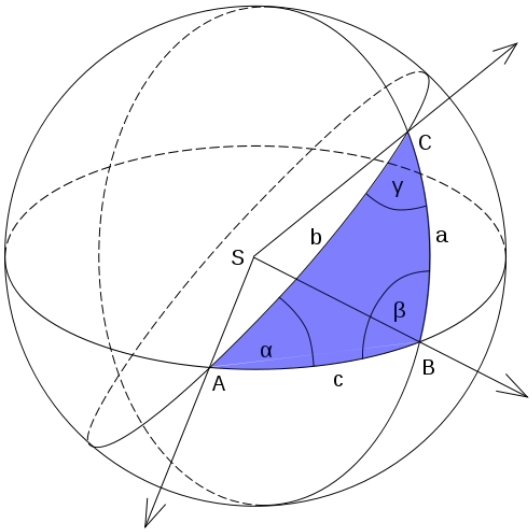
Užívateľsky prívetivá reprezentácia je v klasickom formáte DD°MM,MM. Je teda zrejmé, že v danej triede máme metódu, ktorá konvertuje súradnice z číselnej sekundovej reprezentácie typu long na užívateľsky prívetivú reprezentáciu typu QString.

Z vyššie uvedených viet je zrejmé, že pri zmenách reprezentácií súradníc dochádza často aj k typovým konverziám.

6.3. Sférická trigonometria

Do tejto oblasti som musel nahliadnuť hlavne kvôli logike, ktorá je potrebná pri vykresľovaní bodov a spracovaní mapových podkladov.

Sférická trigonometria je súčasť trigonometrie, ktorá skúma vzťahy medzi prvkami sférického trojuholníka, teda trojuholníka na sférickej ploche. Časť guľovej plochy, ktorá je ohraničená tromi oblúkmi hlavných kružníc, ktoré spojujú tri body guľovej plochy (neležiace na jednej hlavnej kružnici), nazývame **sférickým trojuholníkom** (obrázok 4.).



Obr. 4: Sférický trojuholník[1]

Z tohto trojuholníka sa potom odvodzujú vzorce pre výpočty ortodromy a loxodromy. Ja som vo svojich výpočtoch rátal iba s ortodromou, preto o loxodrome už nebudem ďalej písať.

Ortodroma je najkratšia spojnica dvoch bodov na guľovej ploche. Pre jednotlivé popisné prvky ortodromy a ďalšie výpočty s ňou spojené sa používajú nasledujúce vzorce.

Sférická vzdialenosť: $\sigma = \arccos(\sin\varphi_1 \sin\varphi_2 + \cos\varphi_1 \cos\varphi_2 \cos(\lambda_1 - \lambda_2))$ [3]

Azimut: $\sin\alpha = \frac{(\cos\varphi_2)}{(\sin\sigma)} \sin(\lambda_2 - \lambda_1)$ [3]

Tieto výpočty sa vzťahujú k súradniciam s hodnotami $[\varphi_1,\lambda_1]$ a $[\varphi_2,\lambda_2]$.

6.4. Realizácia vykresľovania mapy

Pri zvážení všetkých okolnosti a toho, aký server respektíve poskytovateľ mapových podkladov bude pre aplikáciu najvýhodnejší som sa rozhodol pre Google maps.

Google maps asi z dôvodu precízneho ajax api, ktoré poskytuje ľahký prístup a ovládanie pomocou javascriptu. Toto riešenie však má jeden problém a to pomerne časté dotazovanie sa servera. Keďže je aplikácia navrhnutá pre mobilné zariadenie bolo treba brat do úvahy rýchlosť pripojenia a tarify za dátové služby. Preto som sa potom rozhodol mapu stohovať ako statický obrázok a žiadať o nový, len v prípade nutnosti prekreslenia a ostatná logika sa prenesie na klienta a nie na Google maps server.

Z praktického hľadiska je tato implementácia navrhnutá spôsobom dokazovania sa serveru pomocou HTTP GET požiadavky, kde v URL priamo nastavujem parametre požadovaného obrázka a súradnice bodu, pre ktorý žiadam danú časť mapy. Je teda zrejmé, že poskytovateľom podkladov máp môže byť akýkoľvek HTTP server poskytujúci na tomto istom princípe funkcie pre prácu s mapami.

Všetka logika je implementovaná na klientovy. Všetko, čo sa na mape má vykresľovať sa prepočítava z GPS súradníc na pixely. Pohyb užívateľa je interpretovaný pohybom mapy, ktorá sa nachádza v jednej spoločnej množine objektov aj s cieľovými bodmi. Užívateľ reprezentovaný bodom so smerovým ukazovateľom sa preto nepohybuje. Tým som docielil toho, že táto komponenta dokáže simulovať pohyb užívateľa na mape.

Samotné vykresľovanie je realizované pomocou balíkov QGraphicsItem, QGraphicsScene a QgraphicsView [2]. K prekresľovaní jednotlivých komponentov dochádza, len v prípadoch, keď je to nutné, alebo je zavolaná metóda update, ktorá neprekresľuje celú komponentu ale iba zadaný štvorec predaný ako parameter. Pretože sú jednotlivé komponenty zgrupované môže takisto dojsť k prekresleniu iba vybranej grupy a nie celej komponenty, čo v oboch prípadoch znižuje nároky na zariadenie.

6.5. Realizácia simulácie kompasu

Kompas, ktorý slúži na ukazovanie smeru pohybu a smeru k zvolenému cieli. Pre časté zmeny smeru pohybu užívateľa som musel implementovať komponentu, ktorá simuluje plynulý pohyb ručičky kompasu ukazujúcej smer k cieli. Túto istú komponentu používam aj pri simulácií pohybu ciferníka, ktorý znázorňuje azimut užívateľa.

Princíp simulácie je založený na rozdelení azimutu na menšie dieliky a ich postupná inkrementácia až kým sa nedosiahne príslušná hodnota. Pri inkrementácií hodnoty azimutu sa emituje signál s parametrom, do ktorého je predaná hodnota azimutu, ten je následne pripojený na slot, v ktorom sa zavolajú príslušné metódy na prekreslenie kompasu. Signál inkrementácie je emitovaný z metódy timerEvent, ktorá je volaná periodicky od časovača podľa zvoleného času. Po skončení simulácie sa časovač zastaví a spustí sa až v prípade, že je zavolaný slot, ktorý zodpovedá za zmenu azimutu.

Samotné vykreslenie je pomerne jednoduché. Do plátna vykresľujem jednotlivé náležitosti potrebné pre vizualizáciu kompasu a podľa nastaveného uhlu pootočenia dochádza, k rotácií ručičky a ciferníku. Jediné, čo som zvažoval bolo to, aké efektívne je používať takto nízky prístup, pretože Qt poskytuje vo svojom frameworku balíky, ktoré programátorovi uľahčia jeho vývoj. Pri porovnávaní týchto dvoch typov vykresľovaní nedochádzalo k veľkým výkonovým rozdielom. Tie by sa prejavili až v prípade, že by som v kompase vykresľoval nejaký obrázok a ten by bolo potrebné napríklad rotovať. V tom prípade by som asi siahol po balíkoch QGraphicsItem, QGraphicsScene a QgraphicsView [2].

Komponenta kompasu reaguje na zmenu súradníc, ktoré sú emitované signálom onGPSPositonChange, kde sa predávajú ako parametre. Slot, ktorý je pripojený na tento signál onPositionChange spustí komponentu simulujúcu pohyb.

7. Uplatnené teoretické a praktické znalosti

V priebehu praxe som uplatnil najviac znalostí z kurzov Programovania v C/C++ a Užívateľských rozhraní, kde som sa stretol s návrhom UI v Qt. Taktiež som sa oboznámil s kompletným návrhom implementovanej aplikácie, čo som už poznal z kurzu Úvod do softvérového inžinierstva.

Nie je však jednoznačné a smerodajné, že iba zo znalosťami z týchto dvoch kurzov by som si vystačil. Mnohokrát som bol v situácií, keď sa mi zišli znalosti aj z iných kurzov poprípade moje vlastné skúsenosti, či už z informatiky alebo inej oblasti.

Počas praxe som sa presvedčil, že praktické skúsenosti sú neodmysliteľnou súčasťou a preto je dôležité ich neustále doplňovať a rozširovať.

8. Chýbajúce vedomosti a skúsenosti

Na začiatku praxe mi chýbali hlavne znalosti s prácou vo verzovacích systémoch. Pretože som ešte v tej dobe pracoval v tíme bolo to pre mňa dosť podstatné manko.

Taktiež aj orientácia v operačných systémoch pre mobilné zariadenia a písanie natívneho kódu. Chýbali mi skúsenosti z nastavovaním kompilátoru a prostredia pre kompilovanie kódu pre Windows Mobile v operačnom systéme Windows XP.

Plánovanie vývoja aplikácie pre mňa bolo tiež novinkou aj keď som už mal nejaké teoretické znalosti zo softvérového inžinierstva. Praktické plánovanie som si však odskúšal prvý-krát.

9. Dosiahnuté výsledky a ich zhodnotenie

Úlohy, ktoré som riešil ma vždy niečomu naučili, či už sa to týkalo vecí vyvíjaných pre danú aplikáciu alebo skúseností, ktoré ma niekam posunuli a využijem ich v budúcnosti. V priebehu praxe som poznal aké procesy pri návrhu tak robustných aplikácií sa v tejto firme používajú. Myslím, že táto skúsenosť mi dala lepší rozhľad a porozumenie tomu ako pri návrhu aplikácií postupovať a hlavne to, že dôležitá je tímová spolupráca.

Z hľadiska samotnej implementácie som si hlavne osvojil tvorbu UI za pomoci Qt frameworku a taktiež písanie kódu v jazyku C/C++, nakoľko je nad ním tento framework postavený. Plusom pre mňa je aj to, že sme používali verzovací systém, čo nám uľahčovalo vývoj aplikácie. Pri práci s týmto systémom som pochopil výhody centrálneho ukladania projektu a následnú prácu s lokálnymi kópiami.

Pretože išlo o multiplatformovú záležitosť bolo potrebné kompilovať zdrojové kódy pod Linuxom(Maemo) aj pod Windowsom(XP). Linuxová verzia, ktorú som mal možnosť odskúšať je distribúciou schopnou pracovať aj na mobilných zariadeniach. Vo Windowse však bolo treba nastaviť kompilátor WinCE, ktorý by bol schopný kompilovať kód pre mobilné zariadenia s operačným systémom Windows Mobile.

10. Záver

Odborná prax vo firme mi podľa môjho názoru dala viac skúseností, ako klasická bakalárska práca. Pri písaní klasickej bakalárskej práce by som síce tiež nadobudol nejaké skúsenosti, ale boli by to skúsenosti, len v určitej oblasti a s konkrétnou témou. Moje pôsobenie vo firme Tieto mi dalo nie len skúsenosti v danej oblasti, ale hlavne som poznal, ako to v takej veľkej firme chodí, aké procesy a metodiky sa používajú pre to aby sa práca zefektívňovala.

V dostatočnej miere som sa oboznámil z návrhom aplikácií a tým, ako praktické skúsenosti prevyšujú vo veľkej miere, len teoretické znalosti. Ale musím podotknúť, že jedno bez druhého sa nezaobíde. Bez teoretických znalostí by to boli často-krát, len pokusy naslepo.

Myslím, že celá odborná prax pre mňa bola veľkým prínosom a rozšírením mojich obzorov, či už z hľadiska skúseností, alebo znalostí. V budúcne nadobudnuté znalosti a uprednostňované praktiky určite využijem.

Pretože sa firma Tieto rozhodla zverejniť zdrojové kódy tejto aplikácie aj s jej stručným popisom na serveri Google code, je možné obe tieto časti nájsť na URL adresách **http://greenhills.googlecode.com/svn/trunk/** a **http://code.google.com/p/greenhills/.**

11. Použitá literatúra

[1] *Wikipedia* [online]. 2010 [cit. 2010-04-19]. Dostupné z WWW: <http://cs.wikipedia.org/wiki/Sf%C3%A9rick%C3%A1_trigonometrie/>

[2] *Qt Reference Documentation* [online]. 2010 [cit. 2010-04-19]. Dostupné z WWW: <http://doc.qt.nokia.com/4.6/>

[3] *Wikipedia* [online]. 2010 [cit. 2010-04-19]. Dostupné z WWW: <http://cs.wikipedia.org/wiki/Ortodroma>

[4] *NMEA* [online]. 2008 [cit. 2010-04-20]. NMEA 0183 Standard. Dostupné z WWW: <http://www.nmea.org/content/nmea_standards/nmea_083_v_400.asp>

[5] BLANCHETTE, Jasmin; SUMMERFIELD, Mark. *C++ GUI Programming with Qt 4*. [s.l.] : Prentice Hall, 02/07/2008. 718 s.

12. Prílohy

A. Obsah CD

1. Vlastné spracovanie bakalárskej práce
2. Zdrojové kódy
3. Triedne diagramy vybraných častí